16. Sensoren en MicroPython

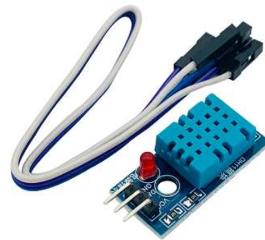
Door de vele GPIO's zijn controllers uitstek geschikt voor het uitlezen van sensoren. In dit deel geven we enkele voorbeelden van sensoren, voornamelijk voor het uitlezen van omgevingsfactoren: luchttemperatuur, vochtigheid, …

Temperatuur en luchtvochtigheid meten met DHT11 en DHT22

De DHT11 en de DHT22 zijn iets oudere sensoren voor temperatuur en luchtvochtigheid. Ze zijn niet erg nauwkeurig maar ze zijn goedkoop, ze worden veel gebruikt en ze zijn heel goed verkrijgbaar. Hier zie je de voornaamste kenmerken:

	DHT11	**DHT22**
Voeding, signaalniveau	3V tot 5V	
Temperatuur	0°C - 50°C ± 2°C	-40°C – 80°C ± 0,5°C
Vochtigheid	20 - 80 %rH ± 5 %	0 – 100 %rH ± 5 %
# metingen / sec.	1	2

Ze zijn verkrijgbaar op een printje met 3 aansluitingen: voeding (3V … 5V), GND en data. Uitgang data sluit je aan op een GPIOpoort.



Figuur 92: DHT11 sensor

Micropython ondersteunt deze sensoren voor verschillende controllers maar niet voor de RP2040. Het programma hieronder is voor de esp8266, de ESP32-c3 en de esp32. Sluit de datapin aan op een willekeurige poort, het nummer van deze poort heb je nodig in de definitie van d in hetprogramma: (d = dht.DHT11 …)

```
#-----------------------------#
#  dht.py    (esp32)          #
#                             #
#  18 december 2021           #
#  Dirk Ghysels               #
```

```
#-----------------------------#
import dht
import machine
from time import sleep

d = dht.DHT11(machine.Pin(16)) # datapin
while True:
    d.measure()
    temp = d.temperature()
    print("temperatuur: ", temp, "°C")
    hum = d.humidity()    # eg. 41 (% RH)
    print("rel. vochtigheid : ", hum, "%rH \n")
    sleep(1)
```

```
Shell
temperatuur:  21 °C
rel. vochtigheid :  64 %rH

temperatuur:  21 °C
rel. vochtigheid :  64 %rH

temperatuur:  21 °C
rel. vochtigheid :  64 %rH
```

Figuur 93: uitlezen van de DHT11

Website how2electronics.com heeft een library en een voorbeeldprogramma gepubliceerd om deze sensor te gebruiken met een andere controllers. De libray bestaat uit één bestand: dht.py. Kopieer dit bestand en sla het op in de controller. Een vereenvoudigde vorm van het voorbeeldprogramma zie je hieronder.

```
#-----------------------------#
#  dht-test.py                #
#-----------------------------#
from machine import Pin, I2C
import utime as time
from dht import DHT11

while True:
    time.sleep(1)
    pin = Pin(28, Pin.OUT, Pin.PULL_DOWN)
    sensor = DHT11(pin)
    try:
        t  = (sensor.temperature)
        h = (sensor.humidity)
    except:
        print("fout")
    else:
        print("Temperature: {}".format(sensor.temperature))
        print("Humidity: {}".format(sensor.humidity))
    time.sleep(1)
```

Het uitlezen van de sensor lukt niet altijd. Om te vermijden dat het programma vastloopt, gebruiken we een try-except-else constructie.

Sluit de datalijn aan op een willekeurige GPIO. In de definitie van pin (regel 9) vervang je 28 door het nummer van de gekozen GPIO.

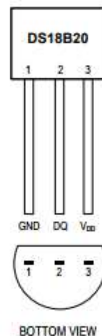
```
Shell
MicroPython v1.17 on 2021-09-02; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
 Temperature: 23.0
 Humidity: 71.0
 Temperature: 23.3
 Humidity: 68.0
 Temperature: 23.3
 Humidity: 70.0
 Temperature: 23.3
 Humidity: 70.0
 Temperature: 23.4
 Humidity: 70.0
```

Figuur 94: metingen van de DHT11

DS18B20: een temperatuursensor

De DS18B20 sensor zet temperatuur om in een 9-bit tot 12-bit getal en heeft een alarmfunctie. Een controller kan meerdere DS18B20's besturen over de 1-Wire bus.

Voeding,	3 V tot 5,5 V
Temperatuur	-55°C tot 125°C ± 0,5°C
resolutie	9 tot 12 bit



Figuur 95: DS18B20

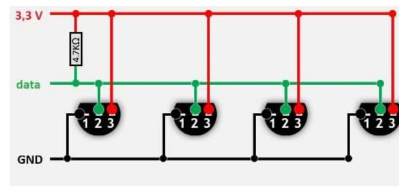
De sensor is verkrijgbaar in verschillende compatibele versies: als los onderdeel in de vorm van een transistor, als een waterdichte sensor met aansluitdraad en als een module met de nodige weerstanden ingebouwd.



Figuur 96: DS18B20: varianten

aansluiting:

Verbind Vcc met 3.3V, GND met GND van de RPI en de datapin met pin 7 van de RPI (fysische én wiringPi nummering). Monteer een weerstand (4700 Ω tot 10 kΩ) tussen de datalijn en de 3,3V lijn. Je kunt meerdere exemplaren parallel aansluiten, ze hebben elk een eigen uniek nummer.



Figuur 97: aansluitschema van een DS18B20

DS18B20 en MicroPython

MicroPython heeft alles in huis om de 1-Wire bus en de DS18B20 te gebruiken. Library onwire ondersteunt de One-Wire bus en library ds18x20 de sensor. Het voorbeeld hieronder toont hoe je de libraries gebruikt. Voor de test hebben we twee sensoren aangesloten op GPIO4. Het programma werkt niet op een SAMD21.

```
#------------------------------#
#  ds18b20.py                  #
#  temperatuur sensor          #
#                              #
#  24 juni 2021                #
#  Dirk Ghysels                #
#------------------------------#
import machine, onewire, ds18x20, utime

dsPin = machine.Pin(4)
ow = onewire.OneWire(dsPin)
sensor = ds18x20.DS18X20(ow)
roms = sensor.scan()
#print('Found a ds18x20 device')
#print(roms)

while True:
    sensor.convert_temp()
    utime.sleep_ms(750)
    k = 1
    for rom in roms:
        print("T%u = %.1f°C" %(k,sensor.read_temp(rom)))
```

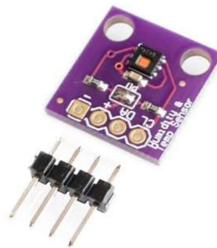
```
        k = k+1
    print('-----------')
    utime.sleep(1)
```

```
MicroPython v1.16 on 2021-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
 T1 = 28.8°C
 T2 = 23.4°C
 -----------
 T1 = 28.9°C
 T2 = 23.4°C
 -----------
 T1 = 28.8°C
 T2 = 23.3°C
 -----------
```

Figuur 98: temperatuur meten met ds18B20

HDC1008: temperatuur en luchtvochtigheid

De **HDC1008** van Texas Instruments is een nauwkeurige en betrouwbare sensor voor temperatuur en luchtvochtigheid. De sensor is enkele mm^2 groot, je kunt er zelf niet aan solderen. We kopen hem als een module op een printje.



Figuur 99: HDC1008

De module geeft de temperatuur in °C, de eenheid voor luchtvochtigheid is %rH. De module werkt met voedingsspanning van 2,7 tot 5,5 volt en is bruikbaar op 3,3 volt systemen en 5 volt systemen. De nauwkeurigheid en snelheid van de HDC1008 zijn veel groter dan die van de DHT11.

	HDC1008
Voeding, signaalniveau	3V tot 5V
Temperatuur	-40°C - 125°C
Nauwkeurigheid temperatuur	0,2°C als 5° < T < 60°
Vochtigheid	0 - 100 %rH
Nauwkeurigheid vochtigheid	1 % als 20% < RH < 60%
conversietijd	6,35 ms

We gebruiken de library van Kai Fricke. Die is gemaakt voor de ESP8266 en het Pyboard, maar werkt ook voor de RP2040. Het demoprogramma moet aangepast worden, hieronder zie je een aangepaste, vereenvoudigde versie:

```
#------------------------------#
#  temp-humi.py                #
#                              #
#  26 maart 2021               #
#  Dirk Ghysels                #
#------------------------------#
import machine
import utime
i2c = machine.I2C(1,sda=machine.Pin(2), scl=machine.Pin(3), freq=400000)
from hdc1008 import HDC1008
print(i2c)
hdc = HDC1008(i2c)
hdc.reset()
while True:
    t = hdc.temp()
    h = hdc.humi()
    #print (t, h)
    print ("temperatuur: %.2f °C" %t)
    print ("luchtvochtigheid: %.2f" %h, "%rH")
    print("------------------------------")
    utime.sleep_ms(1000)
```

We gebruiken twee functies van klasse HDC1008 voor het uitlezen van temperatuur en vochtigheid. We sluiten de aansluiting aan de I2C poort en de 3,3V van de controller.

In volgende programma hebben we de HDC1008 gebruikt om temperatuur en luchtvochtigheid weer te geven op een I2C-LCD met 4 rijen en 20 kolommen. Het LCD gebruikt dezelfde I2C-poort als de sensor.

HDC1008	**RP2040**	**LCD**
GND	GND	GND
3.3 V	3.3 V	
SDA	GP2	SDA
SCL	GP3	SCL
	VBUS	Vcc

```
#------------------------------#
#  temp-humi-lcd.py            #
#                              #
#  26 maart 2021               #
#  Dirk Ghysels                #
#------------------------------#
import machine
import utime
import math
from lcd_api import LcdApi
from pico_i2c_lcd import I2cLcd
I2C_NUM_ROWS = 4
I2C_NUM_COLS = 20
i2c = machine.I2C(1,sda=machine.Pin(2), scl=machine.Pin(3), freq=400000)
lcd = I2cLcd(i2c, 0x27, I2C_NUM_ROWS, I2C_NUM_COLS)
from hdc1008 import HDC1008
```

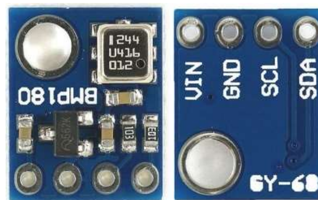
```
hdc = HDC1008(i2c)
hdc.reset()
while True:
    t = hdc.temp()
    h = hdc.humi()
    Tstr = str('T = %.2f' %t)+" 'C"
    Hstr = str('H = %.2f' %h)+" %rH"
    while (len(Tstr)<I2C_NUM_COLS):
        Tstr += " "
    lcd.clear()
    lcd.putstr(Tstr)
    lcd.putstr(Hstr)
    print ("temperatuur: %.2f °C" %t)
    print ("luchtvochtigheid: %.2f" %h, "%rH")
    print("------------------------------")
    utime.sleep_ms(1000)
```

BMP180: temperatuur- en luchtdruk

De BMP180 meet temperatuur en luchtdruk. Luchtdruk is afhankelijk van de hoogte en de weersomstandigheden. De voornaamste kenmerken zijn:

Voeding	1,8 V t3,65,5 V
Luchtdruk bereik Nauwkeurigheid resolutie	300 tot 1100 hPa ± 0,12 hPa 0,01 hPa
Temperatuur bereik Nauwkeurigheid resolutie	-20 °C tot 65 °C ± 0,5 °C 0,1 °C
Interface	I^2C

De BMP180 heeft een moeilijk te solderen SMD-behuizing met aansluitingen onderaan maar wordt ook geleverd als een goedkope module op een printje.



Figuur 100: BMP180 module

De module is eenvoudig aan te sluiten aan de I2C-bus van de RP2040. Sluit Vin aan 3,3 V aan. Het programma gebruikt library bmp180.py van Sebastian Plamaur, zie https://github.com/micropython-IMU/micropython-bmp180. Het programma leest de temperatuur en druk uit de module.

Met variabele bmp180.oversample_sett (0, 1, 2, 3) stel je de nauwkeurigheid in.

0: laagste nauwkeurigheid, snelste meting

3: hoogste nauwkeurigheid, traagste meting

De druk is afhankelijk van de hoogte, met de BMP180 kan je de hoogte berekenen. Variabele bmp180.altitude geeft de hogte als bmp.baseline gelijk is aan de huidige luchtruk op zeeniveau. Die waarde vind je op de website van het KMI (voor de Belgen) of het KNMI (voor de Nederlanders)

```
#-----------------------------#
#  bmp180-test.py             #
#                             #
#  27 maart 2021              #
#  Dirk Ghysels               #
#-----------------------------#
import utime
from machine import Pin, I2C
from bmp180 import BMP180
sda = machine.Pin(2)
scl = machine.Pin(3)
i2c = machine.I2C(1, sda=sda, scl=scl, freq=100000)
bmp180 = BMP180(i2c)
bmp180.oversample_sett = 2
# bmp180.baseline = 101325
while True:
    temp = bmp180.temperature
    p = bmp180.pressure
    # altitude = bmp180.altitude
    print("temperatuur: %.2f °C, luchtdruk: %.2f HPa" %(temp,p/100))
    utime.sleep(1)
```

```
Shell
 temperatuur: 22.37 °C, luchtdruk: 1025.33 HPa
 temperatuur: 22.37 °C, luchtdruk: 1025.40 HPa
 temperatuur: 22.37 °C, luchtdruk: 1025.40 HPa
 temperatuur: 22.37 °C, luchtdruk: 1025.31 HPa
 temperatuur: 22.37 °C, luchtdruk: 1025.31 HPa
 temperatuur: 22.37 °C, luchtdruk: 1025.38 HPa
```

Figuur 101: temperatuur en luchtdruk

Opmerking

- Hou rekening met de beperkte nauwkeurigheid van de metingen. Geen enkele van de geteste meters is nauwkeuriger dan 0,2°C, het heeft geen zin om veel cijfers na de komma af te beelden.
- Geef de sensoren de tijd om warm te lopen. De DS18B20 geeft bij het inschakelen een te hoge waarde. Pas na een tiental minuten is de meting betrouwbaar.
- De nauwkeurigheid van een meter verbetert na callibratie: vergelijk jouw meter met een geijkt toestel en voer in de software correctiefactoren in.